(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 990 986 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
05.04.2000 Bulletin 2000/14

(51) Int Cl.[7]: **G06F 11/07**

(21) Application number: **99306824.6**

(22) Date of filing: **27.08.1999**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **30.09.1998 US 164258**

(71) Applicant: **NCR INTERNATIONAL INC.**
Dayton, Ohio 45479 (US)

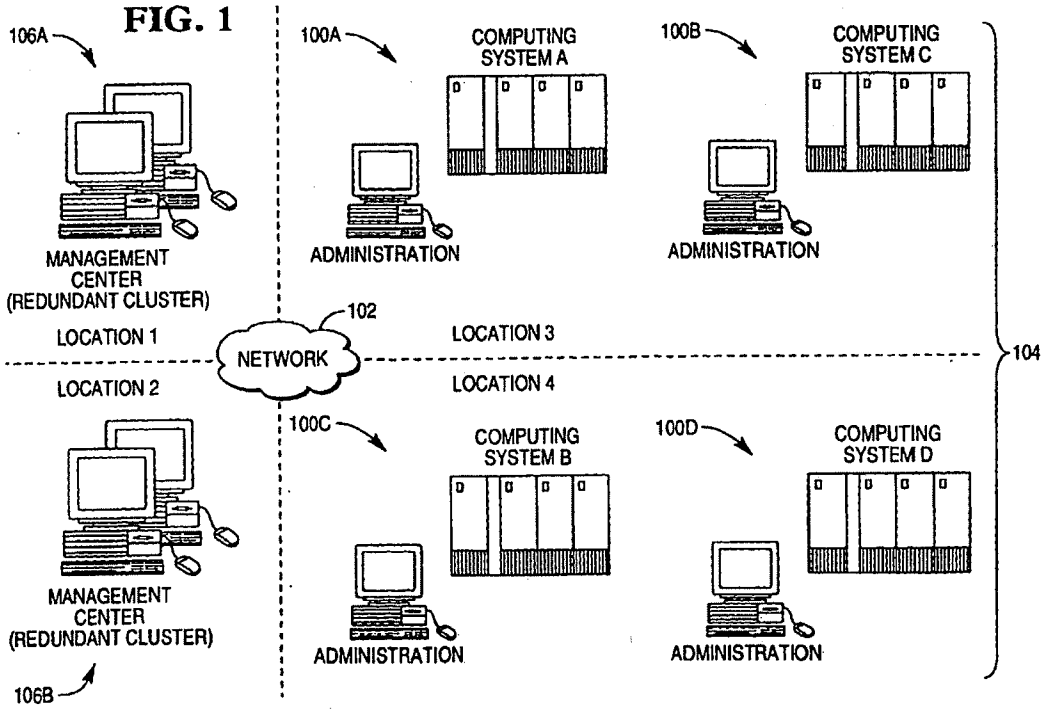
(72) Inventor: **Lynn, Poul Hedegard**
Encinitas, CA 92024 (US)

(74) Representative: **Cleary, Fidelma et al**
International IP Department
NCR Limited
206 Marylebone Road
London NW1 6LY (GB)

(54) Failure recovery of partitioned computer systems including a database schema

(57) A method and apparatus for automatically redistributing tasks to reduce the effect of a computer outage on a computer network. The apparatus comprises at least one redundancy group comprised of one or more computing systems, comprised of one or more computing system partitions. The computing system partition includes copies of a database schema that are replicated at each computing system partition. The redundancy group monitors the status of the computing systems and the computing system partitions, and assigns a task to the computing systems based on the monitored status of the computing systems.

FIG. 1



EP 0 990 986 A2

Description

[0001] The invention relates in general to computer systems, and more particularly, to an automated application fail-over for coordinating applications with database management system (DBMS) availability.

[0002] Many modern computer systems are in nearly continuous use, and have very little time to be taken "down" or "offline" for database updates or preventative maintenance. Further, computer systems increasingly require systems that virtually never fail and have little or no scheduled downtime. As a concurrent requirement, these same systems demand cost-effective computing solutions, open systems to avoid or reduce specific supplier dependencies, and the ability to leverage the latest hardware and software technologies as they become available.

[0003] Modern computer systems also have transitioned from a static installation to a dynamic system that regularly changes. The system continually contains new collections of products and applications that are processing requests from a constantly changing user base. The ability of computing solutions to provide service availability in a dynamic environment is becoming increasingly important, because the pace of change in products and customers' environments is expected to increase. The term "change tolerance" has been used to describe the ability of a computing system to adapt to the dynamic environment required.

[0004] It can be seen, then, that there is a need in the art for a system that provides a high confidence level for continuous processing. It can also be seen, then, that there is a need in the art for a system with a high change tolerance. It can also be seen, then, that there is a need in the art for a system with reasonable development costs and implementation schedules that does not sacrifice the benefits of open systems.

[0005] To overcome the limitations in the prior art described above, and to overcome other limitations that will become apparent upon reading and understanding the present specification, the present invention discloses a method and apparatus for automatically reconfiguring a computer network when a triggering event occurs.

[0006] According to one aspect the invention resides in a failure recovery system, characterized by:

one or more computing systems connected together via a network, wherein each computing system comprises one or more computing system partitions each including at least one copy of a database schema, the copies of the database schema being replicated at each computing system partition within a network;

at least one redundancy group comprised of the computing systems and the computing system partitions, wherein each redundancy group monitors a status of the computing systems and the computing system partitions within the respective redundancy group and assigns a task to the computing systems based on the status of the computing systems and the computing system partitions within the redundancy group.

[0007] The task is preferably database replication within the network.. In one embodiment, the task is assigned to a first one of the computing system that has an available status and is preferably reassigned by the redundancy group to a second one of the computing systems when the status of the first computing system is unavailable.

[0008] The redundancy group may be redefined to include different computing systems.

[0009] The computing system partition may be removed from the redundancy group and may be added to a second redundancy group.

[0010] According to a second aspect, the present invention resides in a method for recovering from a computer failure, characterized by the steps of:

operating one or more computing systems within a network, the computing systems comprising one or more computing system partitions each including at least one copy of a database schema, the copies of the database schema being replicated at each computing system partition within a network;

configuring the computing systems into at least one redundancy group;

monitoring a status of the computing systems and the computing system partitions within the redundancy group; and

assigning a task to the computing systems based on the status of the computing systems and the computing system partitions within the redundancy group.

[0011] The task is preferably a database replication within the network.

[0012] Advantageously, the step of assigning a task is performed when a first one of the computing systems has an available status and the method includes a further step of reassigning a task to a second one of the computing systems when the status of the first one of the computing systems is unavailable.

[0013] The redundancy group may be redefined to include different computing systems.

[0014] The computing system partition may be removed from the redundancy group and may be added to a second redundancy group.

[0015] According to a further aspect, the invention resides in a method for performing tasks within a computer network, characterized by the steps of:

operating one or more computing systems within the computer network, wherein the computing system includes at least one computing system parti-

tion, the computing system partition having at least one copy of a database schema;
configuring the computing systems together via the computer network;
configuring, within the computer network, at least one redundancy group, comprising one or more computing systems and one or more computing system partitions; and
performing at least one task using the computing systems and computing system partitions within the redundancy group.

[0016] For a better understanding of the invention, its advantages, and the objects obtained by its use, reference should be made to the drawings which form a further part hereof, and to the accompanying detailed description, in which there is illustrated and described specific examples in accordance with the invention.

[0017] Referring now to the drawings in which like reference numbers represent corresponding parts throughout:

FIG. 1 is a block diagram that illustrates an exemplary hardware environment that could be used with the present invention;
FIG. 2 further illustrates the components within a computing system of the present invention;
FIG. 3 illustrates the redundancy strategy of the present invention;
FIG. 4 illustrates a model of the computer architecture of the present invention;
FIG. 5 illustrates replication of the database using the present invention;
FIG. 6 illustrates temporal consistency of the database that is propagated by the present invention
FIGS. 7A-7D illustrate the database replication scheme of the present invention; and
FIG. 8 is a flowchart that illustrates exemplary logic performed by the controller according to the present invention.

[0018] In the following description of the preferred embodiment, reference is made to the accompanying drawings which form a part hereof, and in which is shown by way of illustration a specific embodiment in which the invention may be practiced. It is to be understood that other embodiments may be utilized and structural changes may be made without departing from the scope of the present invention.

Overview

[0019] The present invention discloses a method, apparatus, and article of manufacture for distributing computer resources in a network environment to avoid the effects of a failed computing system.

[0020] The apparatus comprises at least one redundancy group comprised of one or more computing systems, comprised of one or more computing system partitions. The computing system partition includes copies of a database schema that are replicated at each computing system partition. The redundancy group monitors the status of the computing systems and the computing system partitions, and assigns a task to the computing systems based on the monitored status of the computing systems.

[0021] Reassignment of a task can occur upon hardware or software problems with the first assignee, or to allow the first assignee to be taken out of service for maintenance purposes. This control is provided by a combination of software systems operating on each of the networked computing systems, and can also be provided on external computing systems called Control Computers. The software on the networked computing system and control computer together determine the status of each of the networked computing systems to determine when to reassign the recipient computing system, and if so, which of the networked computing systems should receive the database updates. The determination is achieved by using periodic messages, time-out values, and retry counts between the software on the networked computing systems and the control computers.

Hardware Environment

[0022] FIG. 1 is an exemplary hardware environment used to implement the preferred embodiment of the invention. The present invention is typically implemented using a plurality of computing systems 100A-100D, each of which generally includes, inter alia, a processor, random access memory (RAM), data storage devices (e.g., hard, floppy, and/or CD-ROM disk drives, etc.), data communications devices (e.g., modems, network interfaces, etc.), etc.

[0023] The computing systems 100A-100D are coupled together via network 102 and comprise a redundancy group 104. Each computing system 100A-D further comprises one or more computing system partitions (not shown), which are described in further detail in FIGS. 2-4. In addition, management centers 106A and 106B can be coupled to network 102. Management centers 106A and 106B are representative only; there can be a greater or lesser number of management centers 106 in the network 102. Further, there can be a greater or lesser number of computing systems 100A-100D connected to the network 102, as well as a greater or lesser number of computing systems 100A-D within the redundancy group 104.

[0024] The present invention also teaches that any combination of the above components, or any number of different components, including computer programs, peripherals, and other devices, may be used to implement the present invention, so long as similar functions are performed thereby. The presentation of the computing system as described in FIG. 1 is not meant to limit

the scope of the present invention, but to illustrate one possible embodiment of the present invention.

Relationships and Operation

[0025] FIG. 2 further illustrates the components within the computing systems 100A-D of the present invention. Within the computing systems 100A-D are one or more computing system partitions (CSPs) 202. Each CSP 202 is coupled to only one copy of a database 204. The computing systems 100A-D are coupled together via network 102.

[0026] Management center computer 106A (or, alternatively, 106B) can be used to control the flow of data from the database copies 204 and updates to the computing systems 100A-100D. The database 204 can also be controlled directly from computing systems 100A-D if desired.

[0027] Each copy of the database 204 is associated with a computing system partition (CSP) 202. As shown in FIG. 2, each computing system 100A-D can have one or more CSP's 202 resident within a computing system, as illustrated in computing system 100A.

[0028] A redundancy group 104 is a collection of CSPs 202 collaborating in an actively redundant fashion on a specific workload using a single replicated database 204 schema. The CSPs 202 may be resident on a single node computing system 100B, C, D, a multi-node computing system 100A, or on selected subsets of computing nodes from one or more multi-node computing systems 100A. Each CSP 202 has an independent database copy of the database 204 for the redundancy group 104. The definition for a CSP 202 is that set of computing resources using a single copy of the replicated database 204.

[0029] The fundamental component of a CSP 202 is a single computing node executing an independent copy of an operating system. However, CSP 202 may consist of multiple nodes and, therefore, multiple operating system instances. The operating system operating on each CSP 202 can be different, e.g., one CSP 202 may be using Windows, while another CSP 202 uses Unix, etc. An operating system instance may be a participant in one and only one redundancy group 104, meaning that the computing nodes comprising a CSP 202 are "owned" by that redundancy group 104. A multi-node computing system 100A can have different nodes participating in different redundancy groups 104, but there must be no overlap between redundancy groups 104.

[0030] To synchronize and replicate the database 204 between the computing systems 100A-100D, one of the computing systems 100A-D is responsible for receiving direct updates of the database 204 via network 102 and disseminating or replicating those updates of database 204 to the remaining computing systems 100A-D.

[0031] As an example, computing system 100B can be designated as the recipient of the direct updates to database 204. Once the updates are received by computing system 100B, computing system 100B then sends a copy of the database 204 with updates to computing systems 100A, 100C, and 100D via network 102. This process continues until computing system 100B has sent a copy of database with updates to all computing systems 100A, C, and D within the network 102.

[0032] If computing system 100B is unavailable, the responsibility of replicating the database and updates shifts to another computing system 100A-D in the network 102. As an example, if computing system 100B is unavailable, the database replication responsibility shifts to computing system 100C, which then receives direct updates. Computing system 100C then replicates the database and updates to computing systems 100A and 100D. Computing system 100C continues the replication until all computing systems 100A and 100D in the network 102 receive copies of the database and updates.

Redundancy Strategy

[0033] FIG. 3 illustrates the hierarchical redundancy strategy of the present invention. To effectively perform the replication of the database 204 and the updates as described in FIG. 2, the present invention partitions the network 102 into redundancy groups 104. Each redundancy group 104 is comprised of computing systems 100A-D, computing system partitions 202, application instances 302, computing system nodes 304, and database copy 306.

[0034] Typical networks 102 have multiple redundancy groups 104. The relationship between redundancy groups 104 is somewhat limited, but all redundancy groups 104 can participate in a global network 102, and a global administration view is typically used for such a network 102. In general, however, redundancy groups 104 are envisioned to be mostly independent of each other and constructed for the purposes of application-level independence, administrative flexibility, or the ability to use computing systems 100A-D of modest capabilities.

[0035] The redundancy group 104 is the fundamental factor of service availability and scalable query performance. The present invention uses the redundancy group 104 to reduce or eliminate a service outage so long as at least one CSP 202 in the redundancy group 104 is fully operational. The present invention also uses the redundancy group 104 to scale query performance beyond that attainable with just one computing system partition 202 and one copy of the database 306. Query performance and availability scale as CSP's 202 are added to a redundancy group 104. With standard computing systems 100A-D, as performance goes up, availability typically goes down. The present invention allows both availability and query performance for computing systems 100A-D to both go up simultaneously.

[0036] Redundancy groups 104 of the present inven-

tion accommodate the condition in which CSPs 202 arbitrarily undergo exit and reintroduction scenarios, but a sufficiently configured redundancy group 104 does not cease proper functionality. The limits of redundancy group 104 functionality and database 204 access is limited by scenarios outside of the control of the computing system 100A-D, e.g., unplanned hardware or software malfunctions, etc.

Computer Architecture Model

[0037] FIG. 4 illustrates a model of the computer architecture of a computing system partition 202 of the present invention. The architecture model 400 has three significant environments: the management environment 402, the run-time environment 404, and the hardware environment 406. The management environment 402 is illustrated as redundancy group management 402. The run-time environment 404 comprises the software components that provide application services directly or indirectly, which is the majority of the components in the model 400. The hardware environment 406 is depicted as the hardware platform, e.g., computer network 102, and peripherals.

[0038] Redundancy group management 402 comprises of the tools, utilities and services necessary to administer, supervise and provide executive control over elements of a redundancy group 104. The components within the redundancy group management 402 environment include redundancy group administration 408, redundancy group supervision 410, redundancy group execution 412.

[0039] The redundancy group administration 408 component provides tools for definition, configuration, and operations of a redundancy group 104. These tools communicate with other tools that provide administrative control of product specific components. Operations include facilities to startup, shutdown, install, and/or upgrade elements of redundancy groups 104. Included in the upgrade and install categories are special facilities necessary for verification. Included in the definition and configuration capabilities are defining policies and procedures to be used by both humans and machines. Additionally, it is foreseen that advanced utilities to determine the scope of failures and subsequently identify recovery procedures would be in this component.

[0040] The redundancy group supervision 410 component provides those services that monitor the health of a redundancy group 104. Included are the services for status request handling, heartbeat setup and monitoring, and failure detection.

[0041] The redundancy group execution 412 component provides those executive services that manage and control the workload of a redundancy group. Included are those services that provide transaction and request-level load balancing and reconfiguration. This component manages and controls the workload of normal transactions as well as recovery requests.

Run-time Environment

[0042] The run-time environment 404 comprises the services necessary to support application programs within redundancy groups 104. The components of the run-time environment 404 include application execution services 414, applications 416, communications resource services 418, global transaction services 420, shared resource services 422, database replication services 424, file i/o 426, remote storage services 428, and network services 430. These components fall into two categories, 1) those components typically utilized by applications 416 directly, and 2) those components typically utilized by applications 416 indirectly. Services that fall into the second category are used by those services in the first category.

[0043] Application execution services 414 provide pre- and post-processing on behalf of an application 416. Such services include application 416 instantiation, parameter marshaling, and queue access services. Application execution services 414 also inform the application 416 of the status of a given transaction request and its disposition; for example, whether it is a normal transaction request, a recovery request, or whether the request is a request to startup or shutdown the application 416. Application execution services 414 also include services necessary to communicate to redundancy group management 402 components. Additionally, application execution services 414 handle application 416 error situations.

[0044] Applications 416 are services to the consumers of a system (network 102), and are composed of software components. Applications 416 are reduced in complexity by leveraging other services in a rich operating environment, such as application 416 execution services 414 and shared resource services 422, since these other services supply needed levels of transparency.

[0045] The communication resource services 418 component comprises services that provide application 416-to-application 416 communications within redundancy groups 104.

[0046] The global transaction services 420 component provides services to maintain transaction context and to coordinate transaction integrity procedures and protocols. These services include facilities for an application 416 to query the global transaction status, and commit or abort transactions.

[0047] The shared resource services 422 component is a general container for services that provide access to shared resources. In a redundancy group 104 the shared resources of interest are replicated databases 204, and, therefore, database 204 access services reside in the shared resource services 422 component. Database 204 access services include services that provide the capability to create, read, write, rewrite, and delete data within a replicated database 204.

[0048] Database replication services 424 fall into the

indirect class of application 416 services. The database replication services 424 propagate database 204 updates transparently to all copies of the database 204 in a redundancy group 104. There are primarily two database 204 replication models, as described in the discussion relating to FIG. 5.

[0049] File i/o services 426 are not utilized directly by customer applications 416, but are provided for use by system software components requiring non-transactional, persistent data storage and access services. File i/o is typically used for logging or journaling functions, event capture, software executables, and data interchange files.

[0050] Remote storage services 428 allow a given file update request to be processed at locations remote from the location of the file i/o request, enabling file replication. System components that take advantage of these services are those that require non-transactional access to queues, logs and system files that would be inappropriate for storage in an database.

[0051] Network services 430 include those services that provide high performance, highly reliable transport of messages. Of specific interest are those services that provide multi-casting of messages which results in an optimal and guaranteed delivery of messages to all destinations in a specified domain of receivers, e.g., computing systems 100A-D. This component also benefits applications indirectly, e.g., customer applications 416 would not call the interface that initiates these services. Rather, these services would be provided to the application 416 through communications resource services 418.

[0052] Network platform 406 is the computing hardware, e.g., network 102, that is used for executing the instructions associated with the application 416, etc.

Database Replication Schemes

[0053] FIG. 5 illustrates replication of the database using the present invention. Within network 424, replication schemes 500 and 502 can be utilized to replicate database 204. Either replication scheme 500 or replication scheme 502, or both, can be used within network 424, depending on the architecture of the redundancy groups 104.

[0054] Database 204 replication is the synchronization mechanism between the database 204 copies in a redundancy group 104. The present invention could also utilize transaction-level replication (reprocessing the entire application transaction on each participating system) instead of entire database 204 replication, but the discussion relating to database 204 replication applies equally well to transaction-level replication. References herein relating to database 204 replication include transaction-level replication.

[0055] At least two distinct database 204 replication models are supported by the present invention, peer/peer replication model 500 and primary/subscriber replication model 502. Other database replication models are envisioned, but the discussion herein is limited to the two models 500 and 502. The peer/peer replication model 502 update transactions are processed on any logical system in a redundancy group 104. Inter-copy database 204 consistency and serializability are maintained either through global network 102 concurrency controls 504, or through commit certifications that occur within the redundancy group 104.

[0056] In the primary/subscriber replication model 502, all update transactions are routed to a single logical system, e.g., computing system 100A, in the redundancy group 104, called the primary system, which propagates updates to the other logical systems, e.g., computing systems 100B-D, after the commitment of a transaction is complete. The update transaction routing is performed transparently and automatically. When the primary logical system, e.g., computing system 100A, exits the redundancy group 104 (for reasons of failure or scheduled downtime) a new primary system is selected. See the discussion relating to FIG. 2.

[0057] FIG. 6 illustrates temporal consistency of the database that is propagated by the present invention. Within either replication model 500 or 502, the database 204 will have temporal inconsistencies because time is required to update the database 204 on each of the network 102 computing systems within a redundancy group 104. Update propagation in replicated database 204 processing has a side effect in that a trade-off must be made between update efficiency and the temporal consistency of the database 204 copies in the redundancy group 104. It is possible to synchronize the database 204 copies by propagating updates before the completion of an update transaction, e.g., before releasing database 204 locks and allowing commit processing to complete. However, absolute synchronization requires propagation protocols that are complex and expensive from a computing perspective.

[0058] The present invention allows the database 204 copies to deviate from each other in a temporal sense, and restrict consistency constraints to serializability and transaction-level atomicity. The approach of the present invention prevents any copy of the database 204 from having "dirty data," "partial updates," or out-of-order updates, but the timing of the appearance of the updates from a given transaction in any particular database 204 copy will be delayed to an unpredictable degree. The temporal deviation between the database 204 copies will be dependent on numerous factors including hardware utilization, instantaneous transaction mix, and network 102 latency.

[0059] The effects of inter-copy temporal inconsistency can be mitigated with numerous application processing techniques, including restriction of updates to selected time windows (during which queries may be restricted), clever partitioning of the query processing workload, and clever partitioning and/or clustering of user queries to specific database copies.

[0060] For a single replicated database schema, shown in replication model 502, each actively redundant configuration will support only one replicated database schema because of transaction-level consistency constraints.

Example Replication Scenario

[0061] FIGS. 7A-7D illustrate the database replication scheme of the present invention. FIG. 7A illustrates network 102 with computing systems 100A-100C. Within computing systems 100A-100C, database 204 is resident, typically on a data storage device.

[0062] As shown in FIG. 7A, data input 700 is received only by computing system 100B. Any of the computing systems 100A-C could be the recipient, but for illustration purposes, computing system 100B is used. Computing system 100B, using DBMS 702, then distributes the data input 700 to computing systems 100A and 100C via network 102.

[0063] This distribution of data input 102 synchronizes the databases 204 that are shared by the network 102. As shown, any of the computing systems 100A-100C can read the data input 700 at terminals 704-708, and use applications 710-714 to process the data stored in database 204.

[0064] FIG. 7B-7D illustrate how the present invention redistributes tasks within the network. FIG. 7B illustrates computing systems 100A-100D. For illustration purposes, computing system 100A is the computing system that is assigned the task of replicating database 204 to the remainder of the computing systems 100B-100D. The task that is assigned to computing system 100A could be a different task, and the computing systems 100B-D that computing system 100A must interact with to complete the task could also be different without deviating from the scope of the invention.

[0065] Computing system 100A replicates the database 204, using the data input 700, to computing system 100B via network path 716. Once that task is complete, computing system 100A replicates the database 204, using the data input 700, to computing system 100C via network path 718. Once that task is complete, computing system 100A replicates the database 204, using the data input 700, to computing system 100D via network path 720. When computing system 100A receives additional data input 700, the process repeats to replicate the changes to database 204 to all the computing systems 100B-100D.

[0066] FIG. 7C illustrates the network when computing system 100A is unavailable. The present invention employs utilities that monitor the status of computing systems 100A-100D that are connected to the network 102. The computing systems 100A-100D are grouped such that the computing systems 100A-100D, when one fails or is unavailable for some other reason, that one of the other computing systems within the group (called a "redundancy group") can take over the tasks that the failed computing system was performing. As an example, when computing system 100A fails or is otherwise unavailable, the present invention reroutes the data input 700 to another computing system in the redundancy group, which, in this case, is computing system 102B. Computing system 102B is assigned the task of replicating database 204, along with the updates to database 204 received via data input 700, to the remaining computing systems 100 in the redundancy group. Computing system 100B replicates the database 204, using the data input 700, to computing system 100C via network path 722. Once that task is complete, computing system 100B replicates the database 204, using the data input 700, to computing system 100D via network path 724.

[0067] FIG. 7D illustrates the network when computing system 100A becomes available again. Once computing system 100A is repaired or is otherwise reconnected to the redundancy group, or, in another example, when a new computing system 100 is added to the redundancy group, computing system 100B continues to perform the task that was assigned to computing system 100B, in this case, the replication of database 204. Computing system 100B, when it performs the replication task, will also replicate the database 204, using the data input 700, to computing system 100A via network path 726.

Logic of the Database Replicator

[0068] FIG. 8 is a flowchart that illustrates exemplary logic performed by the present invention.

[0069] Block 800 represents operating a plurality of computing systems 100A-D within a network, the computing system 100A-D comprising at least one computing system partition including at least one instance of an application, at least one computing system node, and at least one copy of a database schema, the copies of the database schema being replicated at each computing system partition within a network.

[0070] Block 802 represents the computing system 100 configuring the computing systems into at least one redundancy group.

[0071] Block 804 represents the computing system 100 monitoring a status of the computing system and a status of the computing system partition within the redundancy group.

[0072] Block 806 represents the computing system 100 assigning a task to the computing systems based on the status of the computing systems and the status of the computing system partition within the redundancy group.

Conclusion

[0073] This concludes the description of the preferred embodiment of the invention. The following describes some alternative embodiments for accomplishing the

present invention. For example, any type of computer, such as a mainframe, minicomputer, or personal computer, could be used with the present invention. In addition, any software program utilizing (either partially or entirely) a database could benefit from the present invention.

[0074] An apparatus in accordance with the present invention comprises at least one redundancy group comprised of one or more computing systems, which are comprised of one or more computing system partitions. The computing system partition includes copies of a database schema that are replicated at each computing system partition. The redundancy group monitors the status of the computing systems and the computing system partitions, and assigns a task to the computing systems based on the monitored status of the computing systems.

[0075] The foregoing description of the preferred embodiment of the invention has been presented for the purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise form disclosed. Many modifications and variations are possible in light of the above teaching. It is intended that the scope of the invention be limited not by this detailed description, but rather by the claims appended hereto.

Claims

1. A failure recovery system, characterized by:

   one or more computing systems connected together via a network, wherein each computing system comprises one or more computing system partitions each including at least one copy of a database schema, the copies of the database schema being replicated at each computing system partition within a network;
   at least one redundancy group comprised of the computing systems and the computing system partitions, wherein each redundancy group monitors a status of the computing systems and the computing system partitions within the respective redundancy group and assigns a task to the computing systems based on the status of the computing systems and the computing system partitions within the redundancy group.

2. The system of claim 1, wherein the task is a database replication within the network.

3. The system of claim 1, wherein the task is assigned to a first one of the computing system that has an available status.

4. The system of claim 3, wherein the task is reassigned by the redundancy group to a second one of the computing systems when the status of the first computing system is unavailable.

5. The system of claim 1, wherein the redundancy group can be redefined to include different computing systems.

6. The system of claim 1, wherein the computing system partition can be removed from the redundancy group.

7. The system of claim 6, wherein the computing system partition can be added to a second redundancy group.

8. A method for recovering from a computer failure, characterized by the steps of:

   operating one or more computing systems within a network, the computing systems comprising one or more computing system partitions each including at least one copy of a database schema, the copies of the database schema being replicated at each computing system partition within a network;
   configuring the computing systems into at least one redundancy group;
   monitoring a status of the computing systems and the computing system partitions within the redundancy group; and
   assigning a task to the computing systems based on the status of the computing systems and the computing system partitions within the redundancy group.

9. The method of claim 8, wherein the task is a database replication within the network.

10. The method of claim 8, wherein the step of assigning a task is performed when a first one of the computing systems has an available status.

11. The method of claim 10, further comprising the step of reassigning a task to a second one of the computing systems when the status of the first one of the computing systems is unavailable.

12. The method of claim 8, wherein the redundancy group can be redefined to include different computing systems.

13. The system of claim 8, wherein the computing system partition can be removed from the redundancy group.

14. The method of claim 13, wherein the computing system partition can be added to a second redundancy group.

15. A method for performing tasks within a computer network, characterized by the steps of

operating one or more computing systems within the computer network, wherein the computing system includes at least one computing system partition, the computing system partition having at least one copy of a database schema;
configuring the computing systems together via the computer network;
configuring, within the computer network, at least one redundancy group, comprising one or more computing systems and one or more computing system partitions; and
performing at least one task using the computing systems and computing system partitions within the redundancy group.

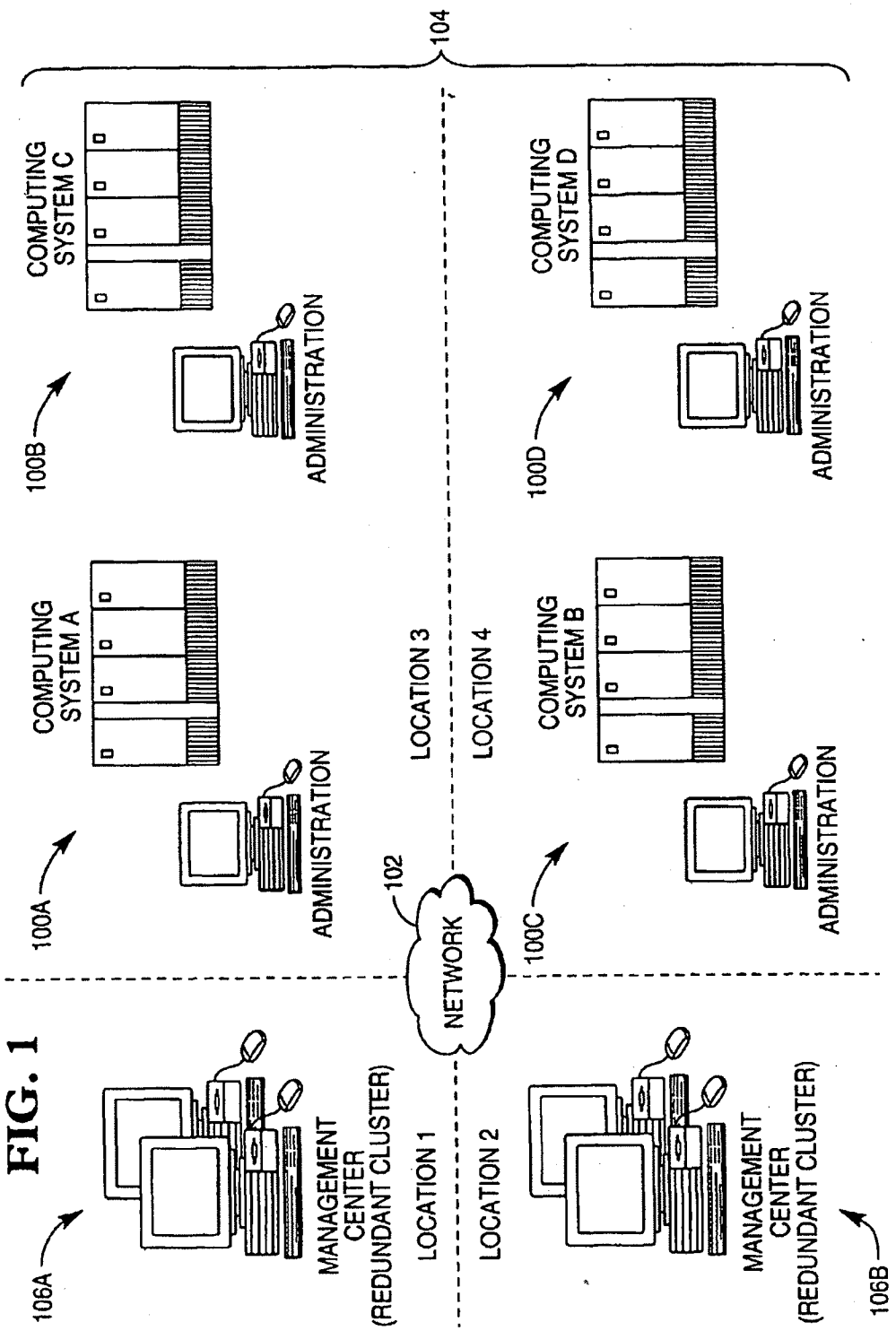

104
COMPUTING SYSTEM C
COMPUTING SYSTEM D
COMPUTING SYSTEM A
COMPUTING SYSTEM B
ADMINISTRATION
ADMINISTRATION
ADMINISTRATION
ADMINISTRATION
100A
100B
100C
100D
LOCATION 3
LOCATION 4
LOCATION 1
LOCATION 2
102
NETWORK
MANAGEMENT CENTER (REDUNDANT CLUSTER)
MANAGEMENT CENTER (REDUNDANT CLUSTER)
106A
106B

FIG. 1

FIG. 2

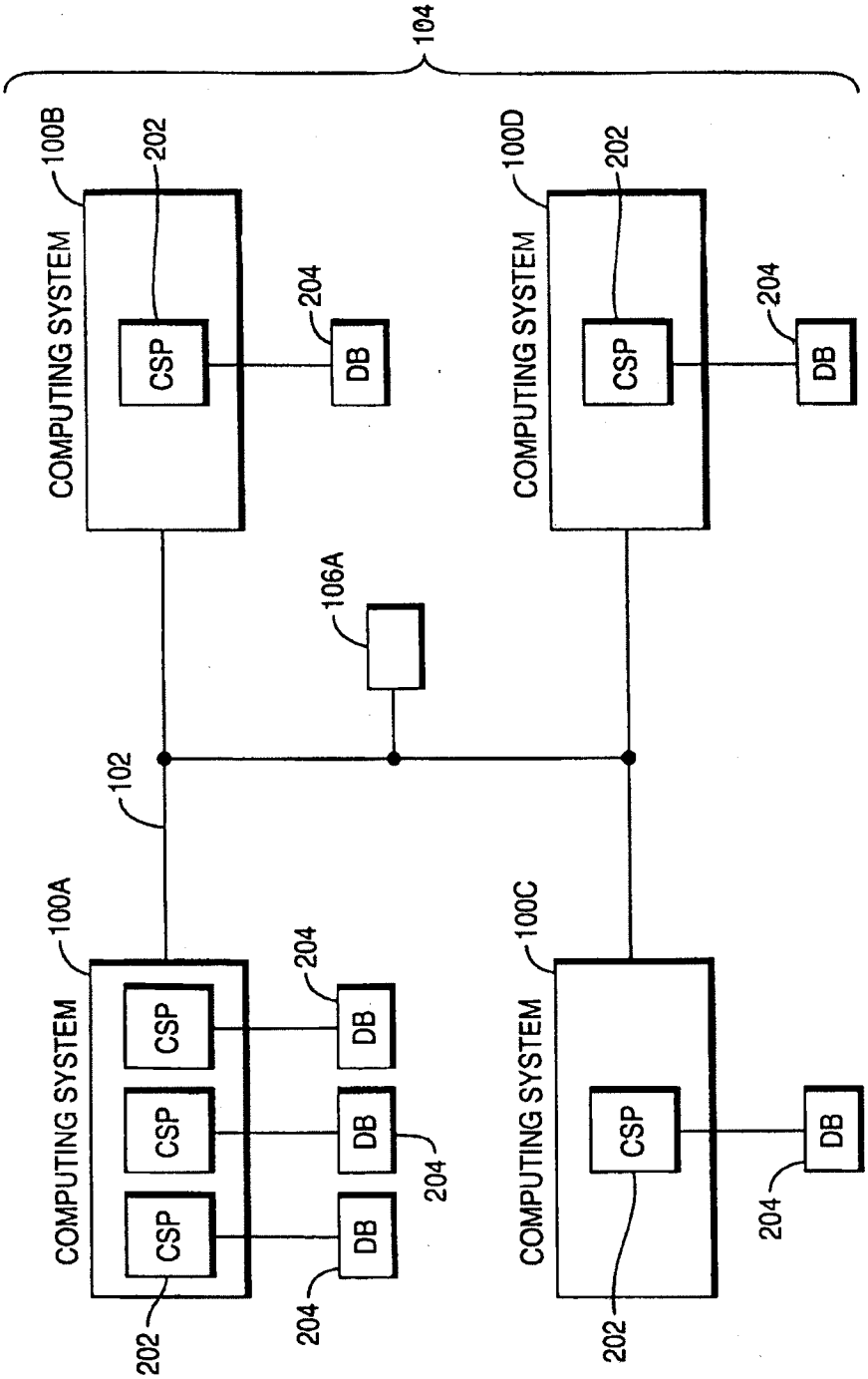
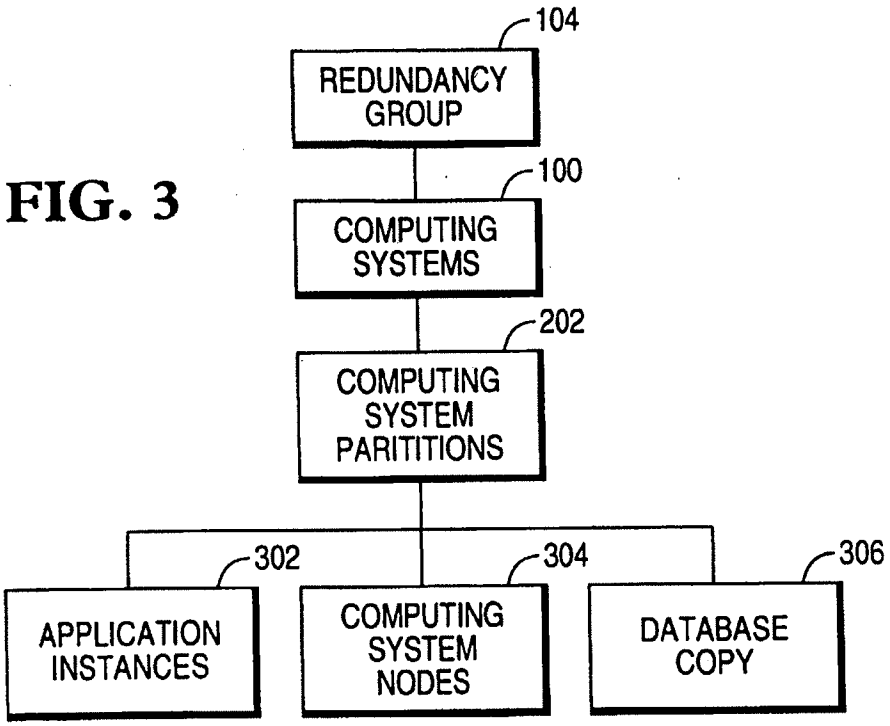

104
100B
COMPUTING SYSTEM
202
CSP
204
DB
100D
COMPUTING SYSTEM
202
CSP
204
DB
106A
102
100A
COMPUTING SYSTEM
CSP
CSP
CSP
204
DB
DB
DB
204
202
204
100C
COMPUTING SYSTEM
CSP
DB
204
202

FIG. 3


104
REDUNDANCY GROUP
100
COMPUTING SYSTEMS
202
COMPUTING SYSTEM PARITITIONS
302
APPLICATION INSTANCES
304
COMPUTING SYSTEM NODES
306
DATABASE COPY

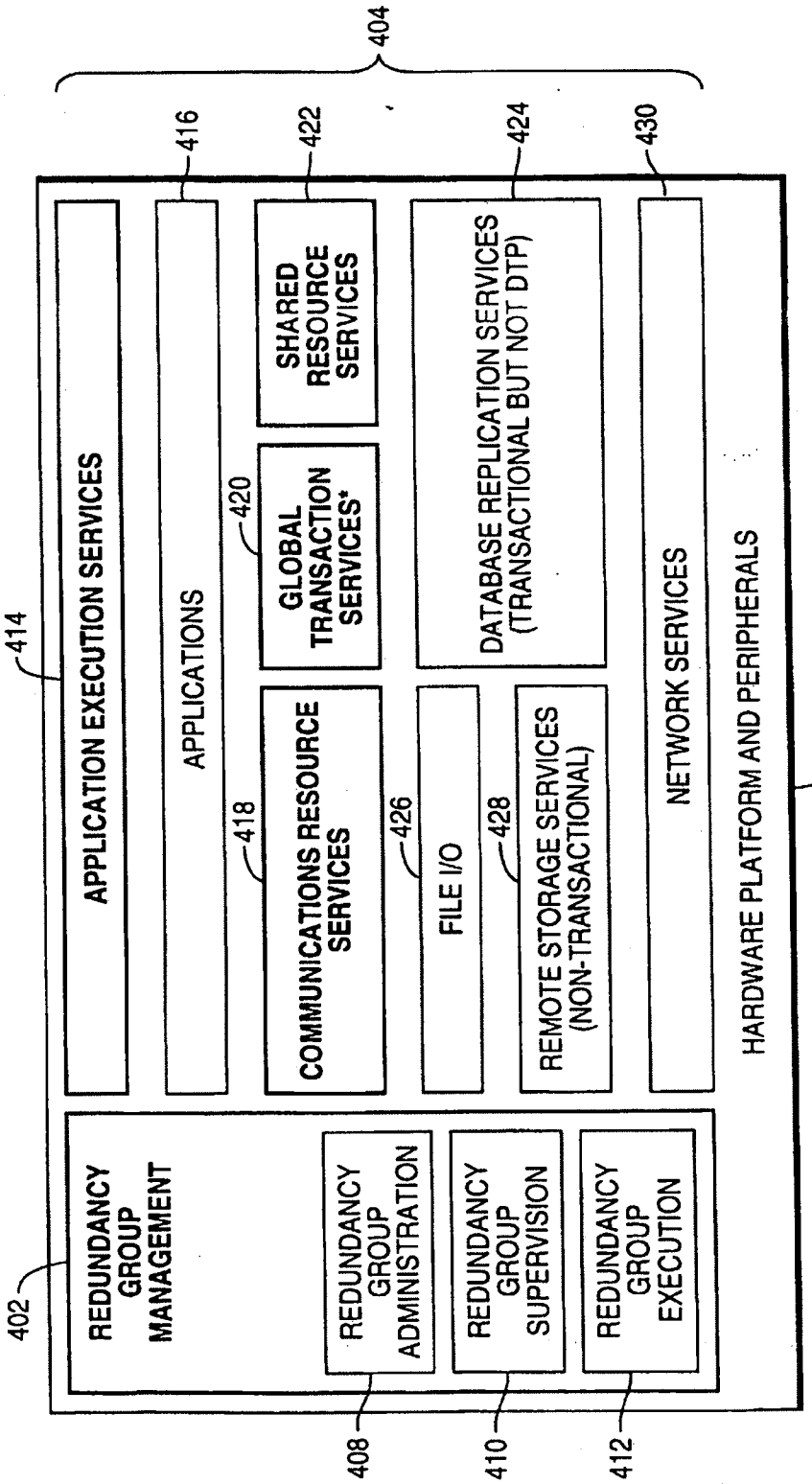

400
404
414
416
422
424
430
420
418
426
428
406
402
408
410
412
APPLICATION EXECUTION SERVICES
APPLICATIONS
SHARED RESOURCE SERVICES
GLOBAL TRANSACTION SERVICES*
DATABASE REPLICATION SERVICES (TRANSACTIONAL BUT NOT DTP)
COMMUNICATIONS RESOURCE SERVICES
FILE I/O
REMOTE STORAGE SERVICES (NON-TRANSACTIONAL)
NETWORK SERVICES
HARDWARE PLATFORM AND PERIPHERALS
REDUNDANCY GROUP MANAGEMENT
REDUNDANCY GROUP ADMINISTRATION
REDUNDANCY GROUP SUPERVISION
REDUNDANCY GROUP EXECUTION

FIG. 4

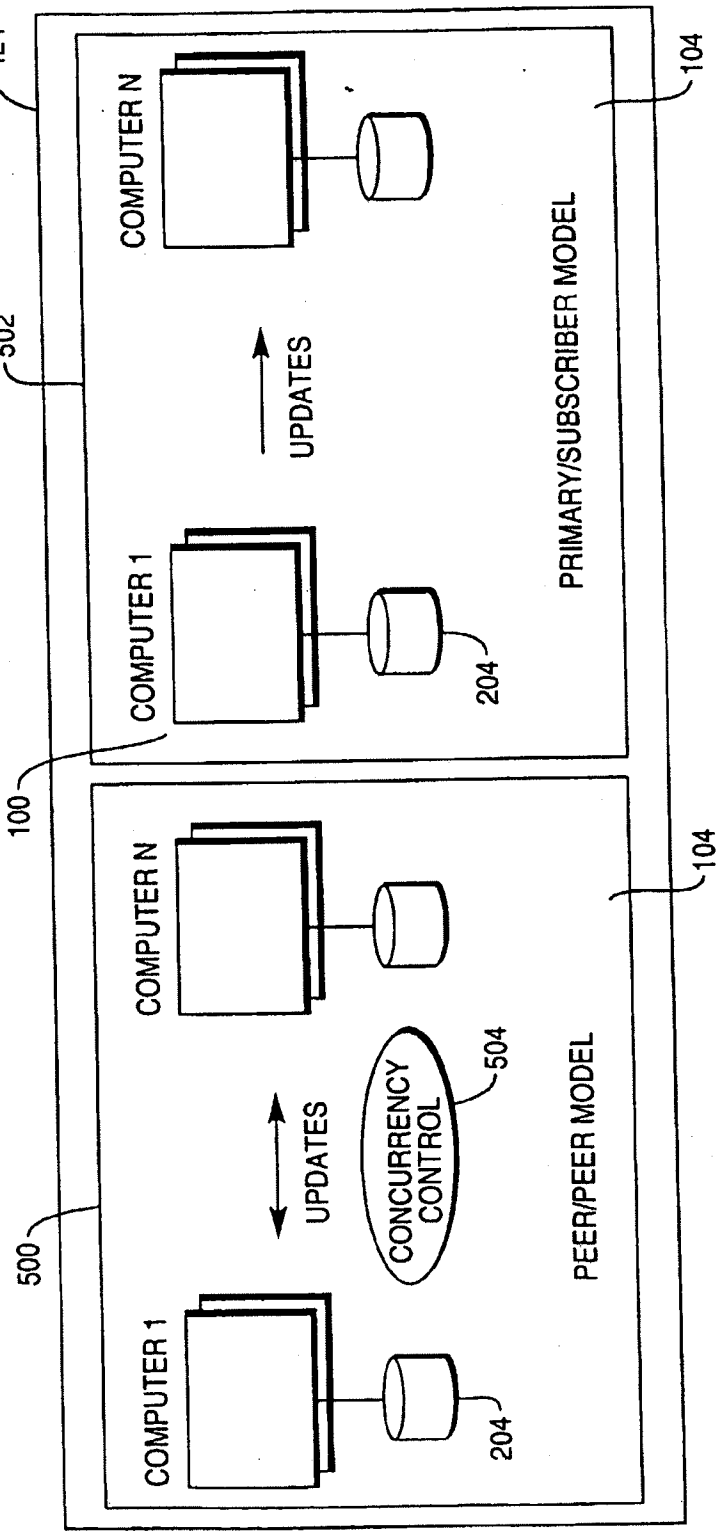
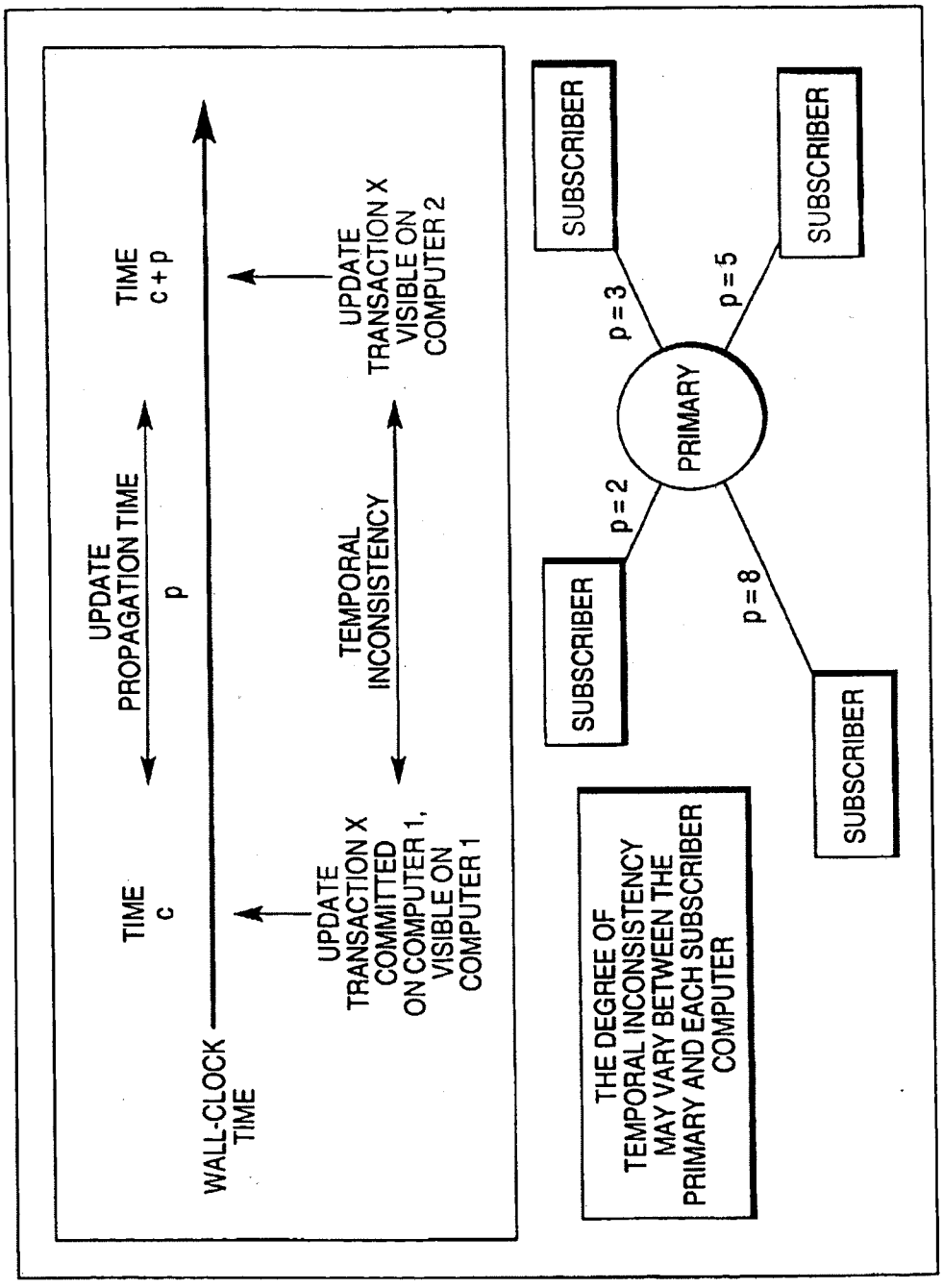

424
COMPUTER N
104
502
UPDATES
PRIMARY/SUBSCRIBER MODEL
COMPUTER 1
204
100
104
COMPUTER N
504
CONCURRENCY CONTROL
UPDATES
500
PEER/PEER MODEL
COMPUTER 1
204

FIG. 5

FIG. 6


WALL-CLOCK TIME
TIME c
TIME c + p
UPDATE PROPAGATION TIME
p
UPDATE TRANSACTION X COMMITTED ON COMPUTER 1, VISIBLE ON COMPUTER 1
TEMPORAL INCONSISTENCY
UPDATE TRANSACTION X VISIBLE ON COMPUTER 2
PRIMARY
SUBSCRIBER
SUBSCRIBER
SUBSCRIBER
SUBSCRIBER
p = 2
p = 3
p = 5
p = 8
THE DEGREE OF TEMPORAL INCONSISTENCY MAY VARY BETWEEN THE PRIMARY AND EACH SUBSCRIBER COMPUTER

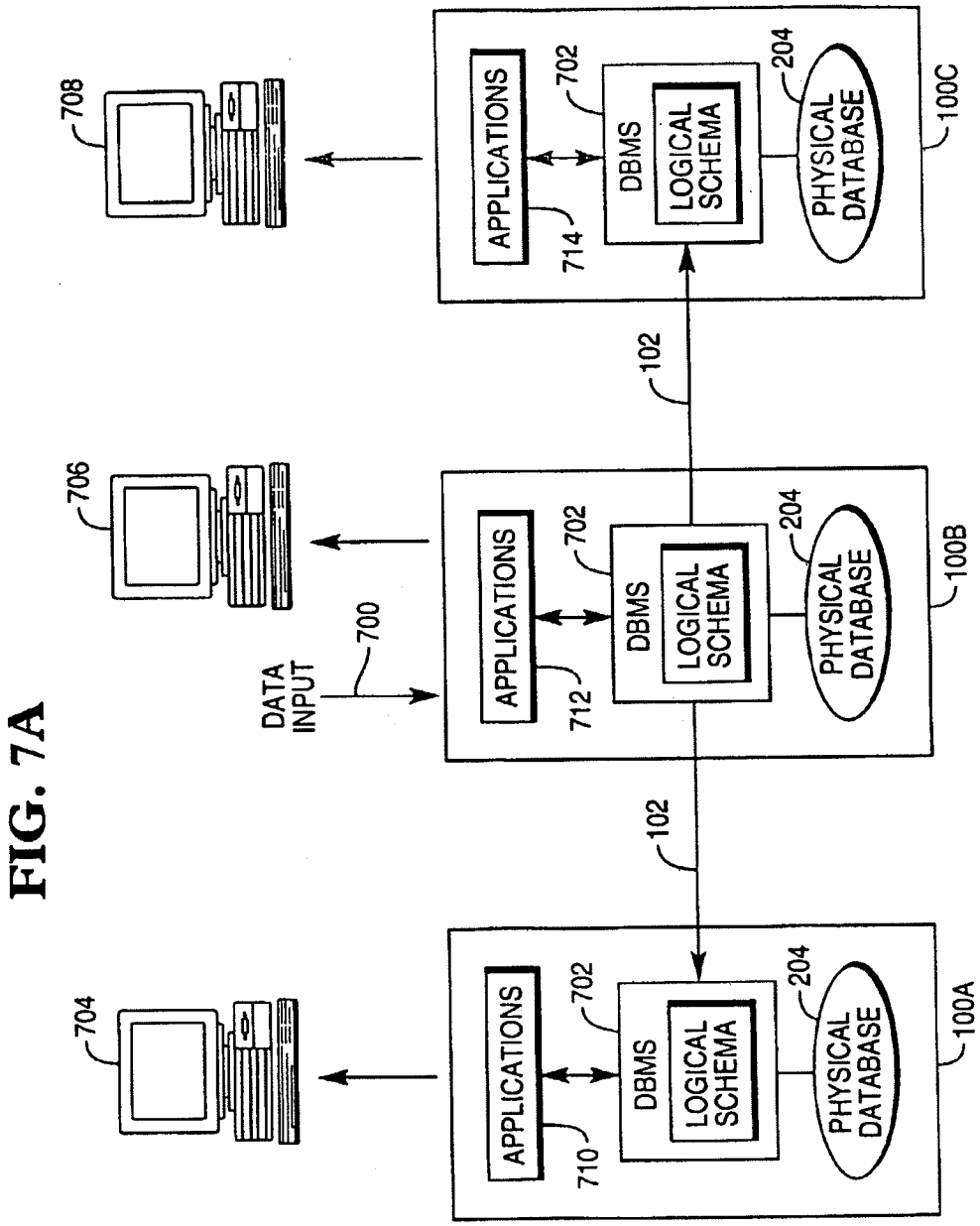

708
APPLICATIONS
702
DBMS
LOGICAL SCHEMA
204
PHYSICAL DATABASE
100C
714
102
706
DATA INPUT
700
APPLICATIONS
702
DBMS
LOGICAL SCHEMA
204
PHYSICAL DATABASE
100B
712
102
704
APPLICATIONS
702
DBMS
LOGICAL SCHEMA
204
PHYSICAL DATABASE
100A
710

FIG. 7A

FIG. 7B

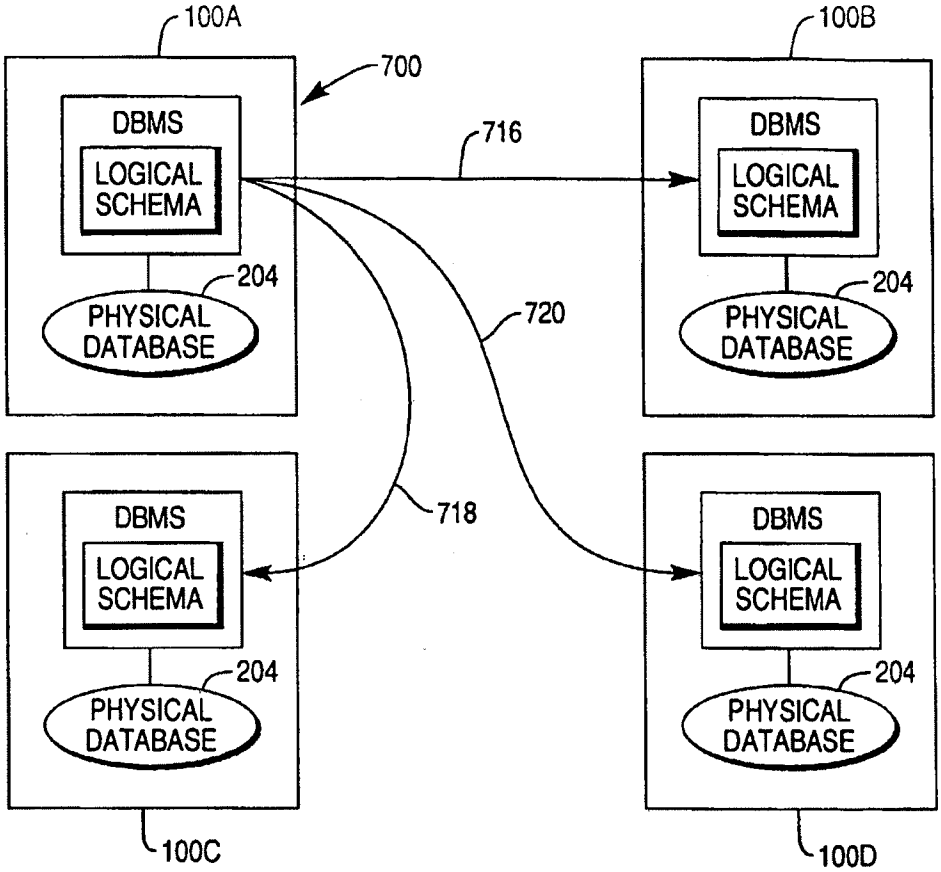

100A
700
DBMS
LOGICAL SCHEMA
204
PHYSICAL DATABASE
716
100B
DBMS
LOGICAL SCHEMA
204
PHYSICAL DATABASE
720
718
DBMS
LOGICAL SCHEMA
204
PHYSICAL DATABASE
100C
DBMS
LOGICAL SCHEMA
204
PHYSICAL DATABASE
100D

FIG. 7C

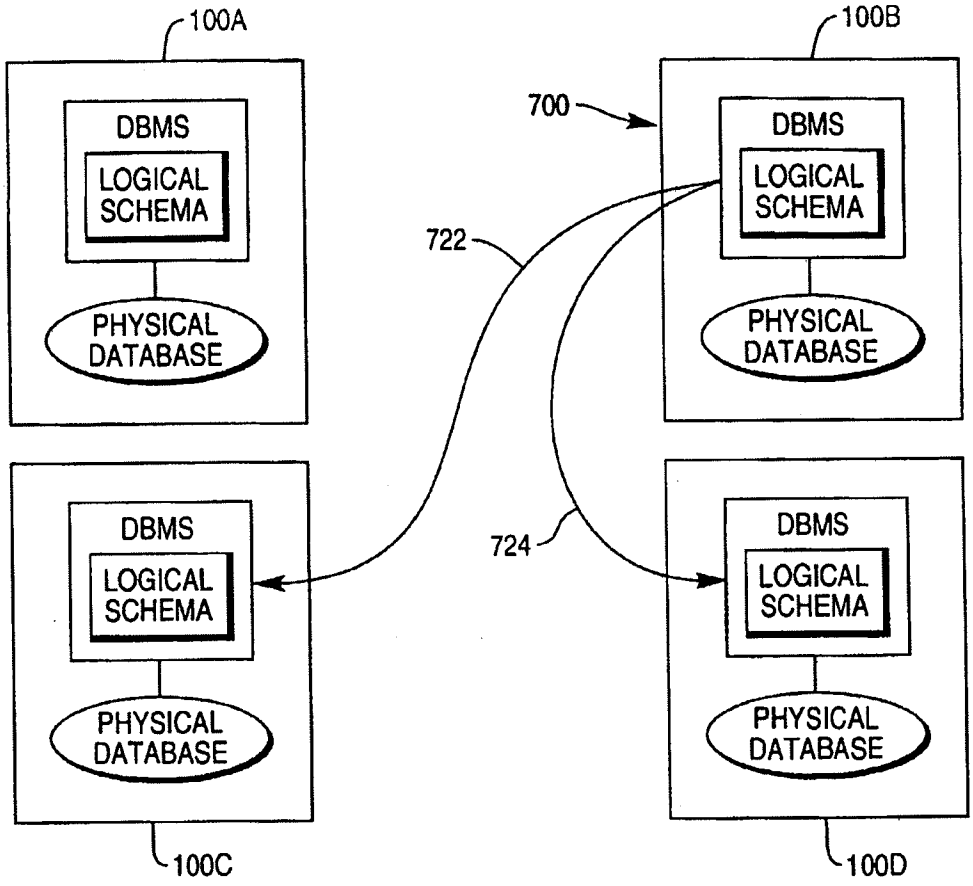

100A
DBMS
LOGICAL SCHEMA
PHYSICAL DATABASE
100B
700
DBMS
LOGICAL SCHEMA
PHYSICAL DATABASE
722
724
DBMS
LOGICAL SCHEMA
PHYSICAL DATABASE
100C
DBMS
LOGICAL SCHEMA
PHYSICAL DATABASE
100D

FIG. 7D

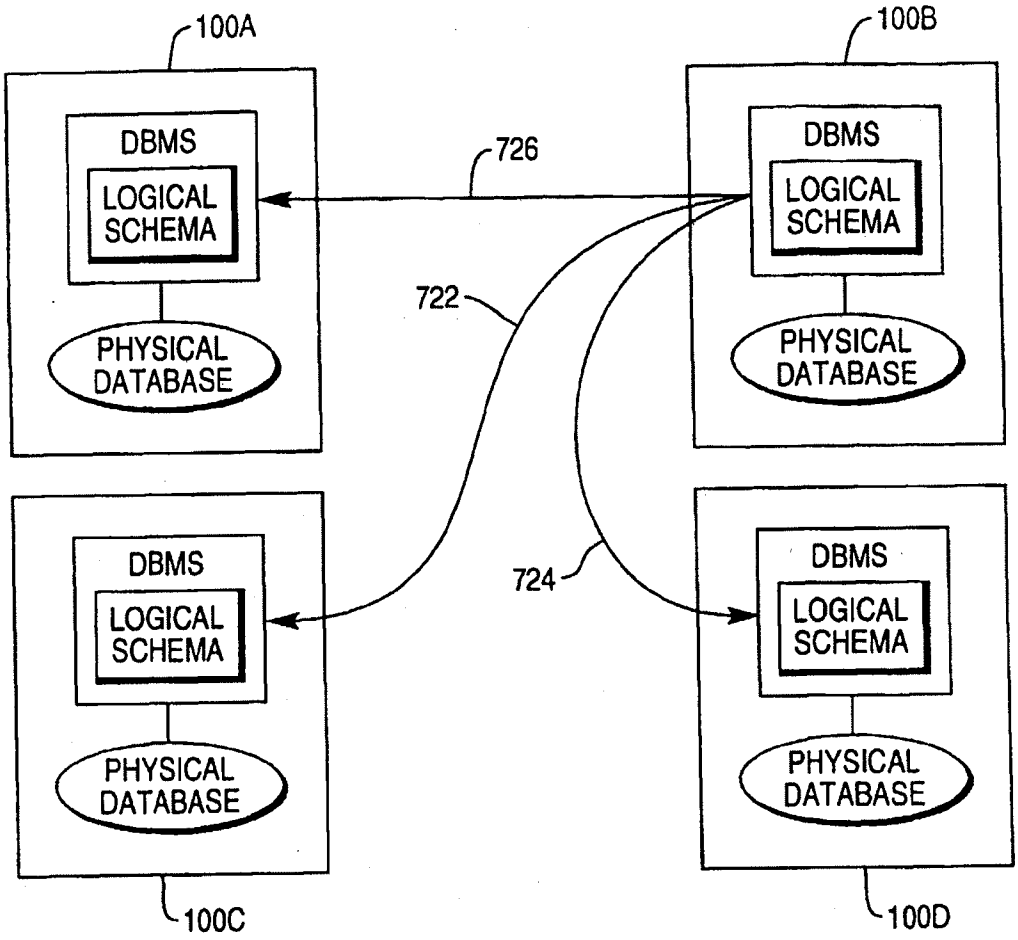
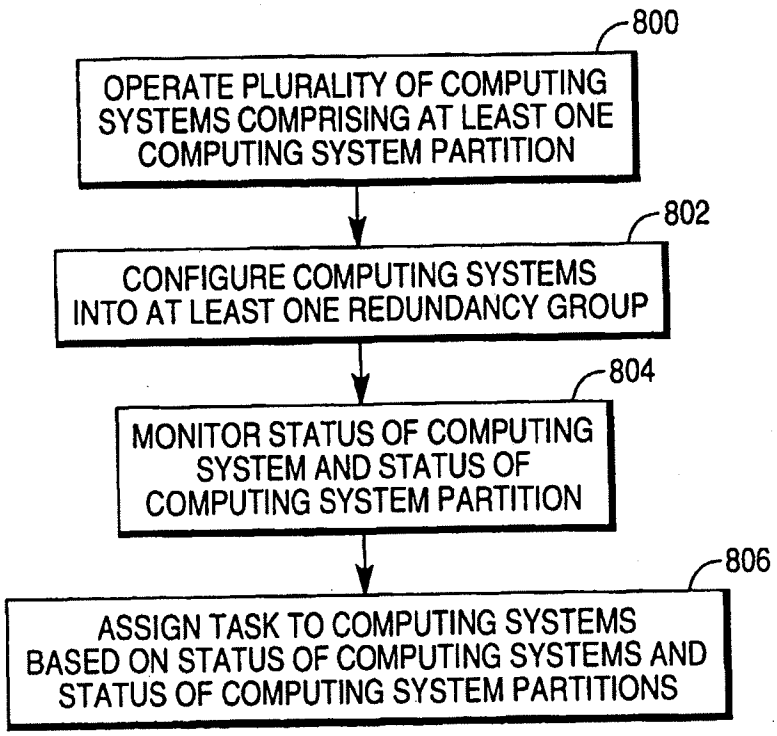

100A
100B
DBMS
LOGICAL SCHEMA
PHYSICAL DATABASE
726
722
724
100C
100D

FIG. 8


800
OPERATE PLURALITY OF COMPUTING SYSTEMS COMPRISING AT LEAST ONE COMPUTING SYSTEM PARTITION
802
CONFIGURE COMPUTING SYSTEMS INTO AT LEAST ONE REDUNDANCY GROUP
804
MONITOR STATUS OF COMPUTING SYSTEM AND STATUS OF COMPUTING SYSTEM PARTITION
806
ASSIGN TASK TO COMPUTING SYSTEMS BASED ON STATUS OF COMPUTING SYSTEMS AND STATUS OF COMPUTING SYSTEM PARTITIONS